What is the next step?

- We are going to normalize.
  - 1st NF: no multivalued attributes.
  - 2nd NF: functional dependencies.
    - Either only one PK exists
    - No non-key attribute exist
    - Or all non-key attribs depend on the full set of PK attribs.
  - 3rd NF
    - No transitive dependencies: dependencies between NON KEY ATTRIBUTES
- We are going to implement.
  - mySQL software

Chapter 6

SQL

SQL Is:

- Structured Query Language
- The standard for relational database management systems (RDBMS)
- SQL-92 Standard -- Purpose:
  - Specify syntax/semantics for data definition and manipulation
  - Define data structures
  - Enable portability
  - Allow for later growth/enhancement to standard

Benefits of a Standardized Relational Language

- Reduced training costs
- Productivity
- Application portability
- Application longevity
- Reduced dependence on a single vendor
- Cross-system communication

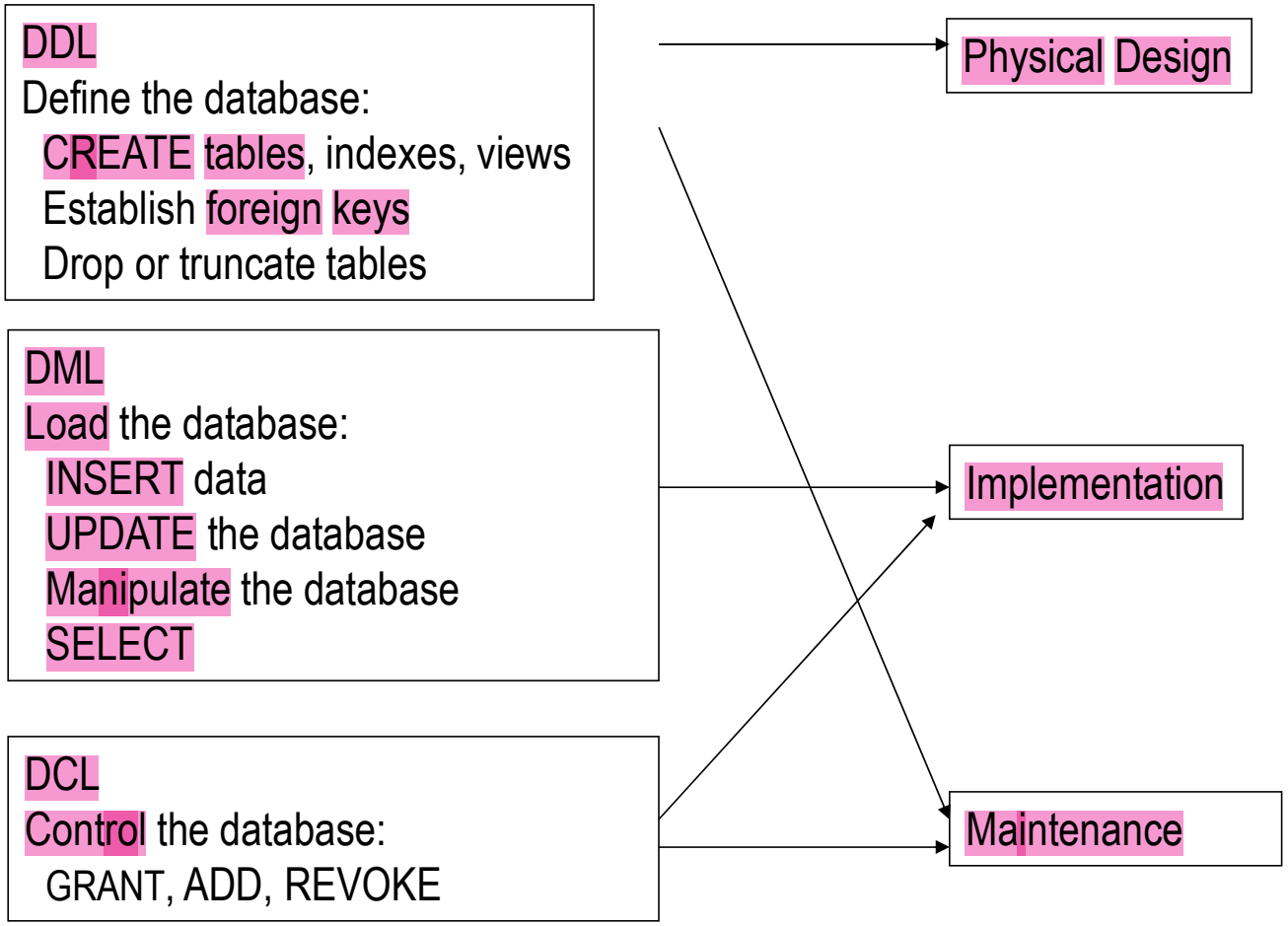
SQL Data types (from Oracle9i)

- String types
  - CHAR(n) – fixed-length character data, n characters long Maximum length = 2000 bytes
  - VARCHAR2(n) – variable length character data, maximum 4000 bytes
  - LONG – variable-length character data, up to 4GB. Maximum 1 per table
- Numeric types
  - NUMBER(p,q) – general purpose numeric data type
  - INTEGER(p) – signed integer, p digits wide
  - FLOAT(p) – floating point in scientific notation with p binary digits precision
- Date/time type
  - DATE – fixed-length date/time in dd-mm-yy form

SQL commands classified

- **Data Definition Language (DDL):**
  - Commands that define a database, including creating, altering, and dropping tables and establishing constraints.
- **Data Manipulation Language (DML)**
  - Commands that maintain and query a database.
- **Data Control Language (DCL)**
  - Commands that control a database, including administering privileges and committing data.

DDL, DML, DCL, and the database development process



DML

- Most frequently used Statements are DMLs
- Used for Populating tables with data
  - INSERT
- Querying tables
  - SELECT
  - MODIFY OR UPDATE
  - DELETE
  - MAKE TABLE
  - APPEND TABLE

Insert Statement

- Adds data to a table
- Inserting into a table
  - INSERT INTO CUSTOMER VALUES (001, 'CONTEMPORARY Casuals', 1355 S. Himes Blvd.', 'Gainesville', 'FL', 32601);
- Inserting a record that has some null attributes requires identifying the fields that actually get data
  - INSERT INTO PRODUCT (PRODUCT_ID, PRODUCT_DESCRIPTION,PRODUCT_FINISH, STANDARD_PRICE, PRODUCT_ON_HAND) VALUES (1, 'End Table', 'Cherry', 175, 8);
- Inserting from another table
  - INSERT INTO CA_CUST SELECT * FROM CUSTOMER WHERE STATE = 'CA';

Update Statement

- Modifies data in existing rows

- UPDATE PRODUCT SET UNIT_PRICE = 775
  WHERE PRODUCT_ID = 7;

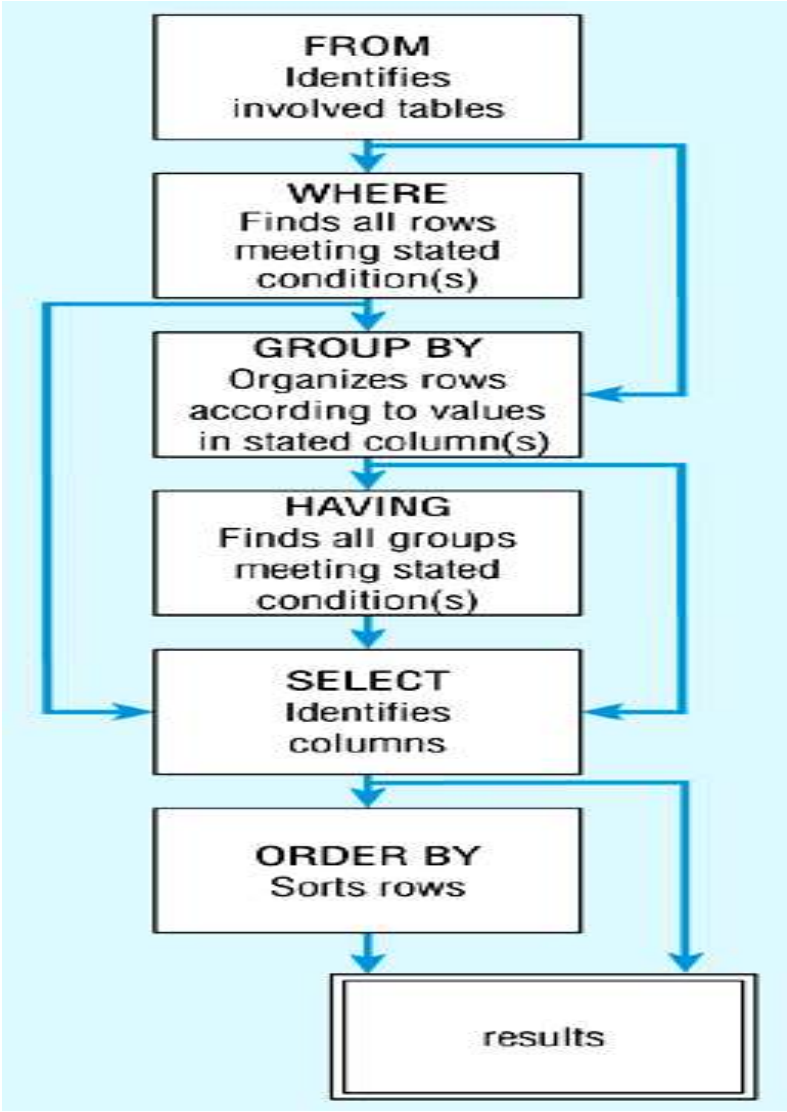
Delete Statement

- Removes rows from a table
- Delete certain rows
  - DELETE FROM CUSTOMER WHERE STATE = ‘HI’;
- Delete all rows
  - DELETE * FROM CUSTOMER;
  - DROP CUSTOMER;

The SELECT Statement

- Used for queries on single or multiple tables
- Clauses of the SELECT statement:
  - SELECT
    - List the columns (and expressions) that should be returned from the query
  - FROM
    - Indicate the table(s) or view(s) from which data will be obtained
  - WHERE
    - Indicate the conditions under which a row will be included in the result
  - GROUP BY
    - Indicate categorization of results
  - HAVING
    - Indicate the conditions under which a category (group) will be included
  - ORDER BY
    - Sorts the result according to specified criteria

SQL statement processing order (adapted from van der Lans, p.100)



SELECT Example

- Find products with standard price less than $275

- SELECT PRODUCT_NAME, STANDARD_PRICE
- FROM PRODUCT
- WHERE STANDARD_PRICE < 275

Comparison Operators in SQL

Table 7-3 Comparison Operators in SQL

Operator	*Meaning*
=	Equal to
>	Greater than
> =	Greater than or equal to
<	Less than
< =	Less than or equal to
<>	Not equal to
! =	Not equal to

SELECT Example – Boolean Operators

AND, OR, and NOT Operators for customizing conditions in WHERE clause

```
SELECT PRODUCT_DESCRIPTION,
PRODUCT_FINISH, STANDARD_PRICE
FROM PRODUCT
WHERE (PRODUCT_DESCRIPTION LIKE '%Desk'
OR PRODUCT_DESCRIPTION LIKE '%Table')
AND UNIT_PRICE > 300;
```

Note: the LIKE operator allows you to compare strings using wildcards. For example, the % wildcard in '%Desk' indicates that all strings that have any number of characters preceding the word "Desk" will be allowed

SELECT Example
Using a Function

- Using the COUNT ***aggregate function*** to find totals

- SELECT COUNT(*) FROM ORDER_LINE
  WHERE ORDER_ID = 1004;